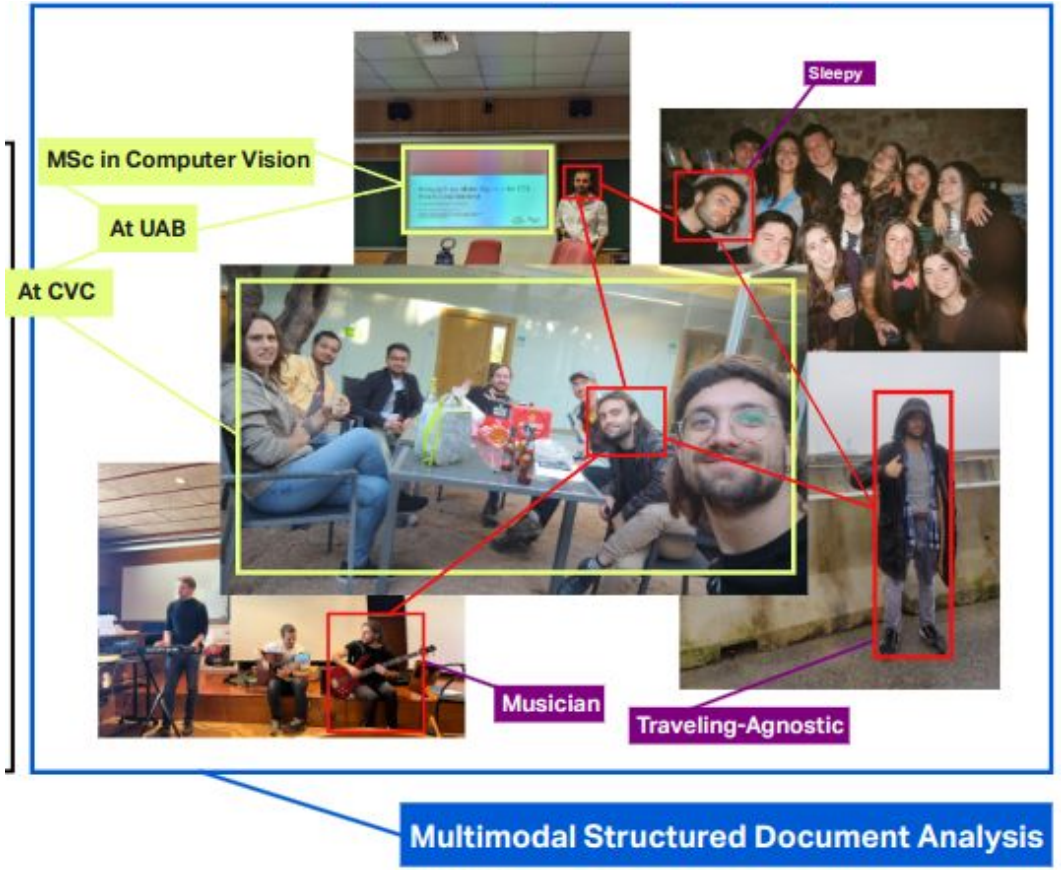
Visual Model Checking: Graph-Based Inference of Visual Routines for Image Retrieval

Adrià Molina Rodríguez, Oriol Ramos Terrades
Josep Lladós Canet

Computer Vision Center

ICPR 2026 International Conference on Pattern Recognition LYON

Table Of Contents



This work has been partially supported by the Spanish project PID2024-157778OB-I00, Ministerio de Ciencia e Innovación, the Departament de Cultura of the Generalitat de Catalunya, and the CERCA Program / Generalitat de Catalunya. Adrià Molina is funded with the PRE2022-101575 grant provided by MCIN / AEI / 10.13039 / 501100011033 and by the European Social Fund (FSE+).

Preface and Motivation

At the limits of embedding-based image retrieval.

While CLIP-alike systems excel in zero-shot image retrieval, through our experiments, we identified three main pitfalls:

Attribute Dominance

(street, is, snowy)

“A man skating in a snowy street”

Counting and Enumeration

(plate, has, meat)

(plate, has, rice)

“A plate with meat, rice and vegetables”

Reading text and Symbols

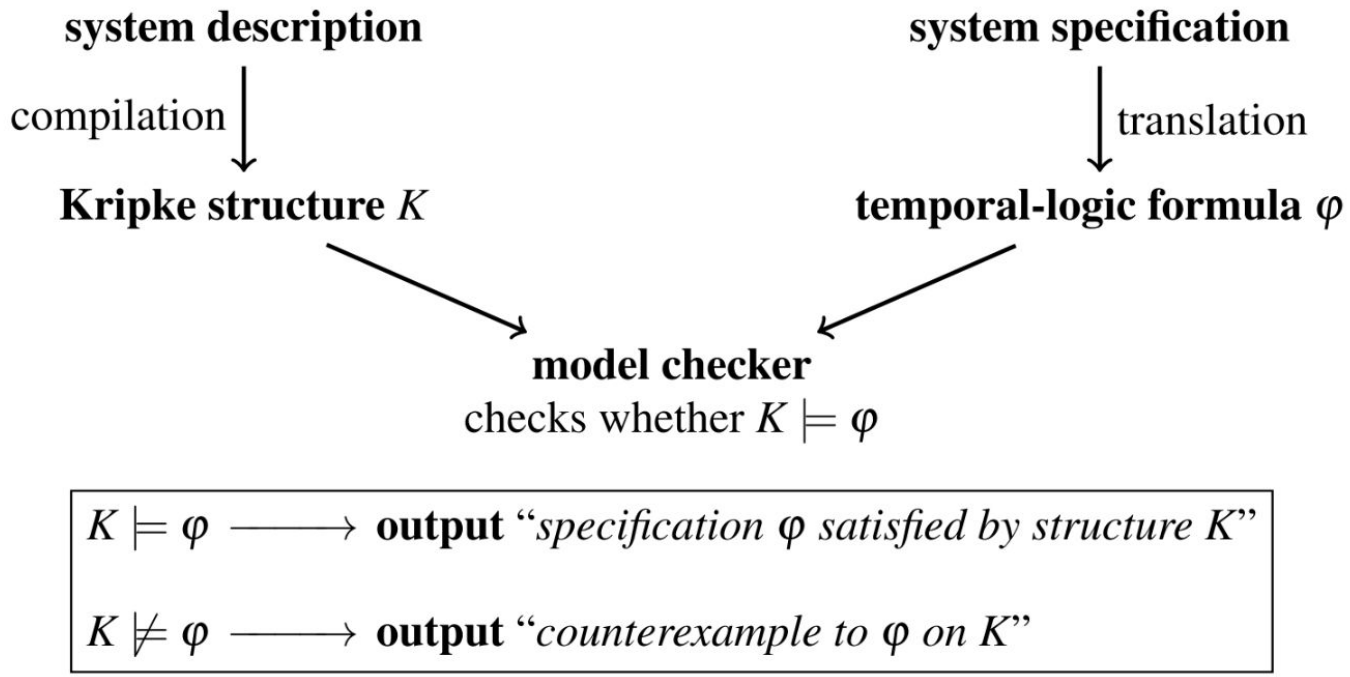
((street, has, sign), is, one-way)

“Many people walking by a one-way sign”

Visual Model Checking

What is model checking?

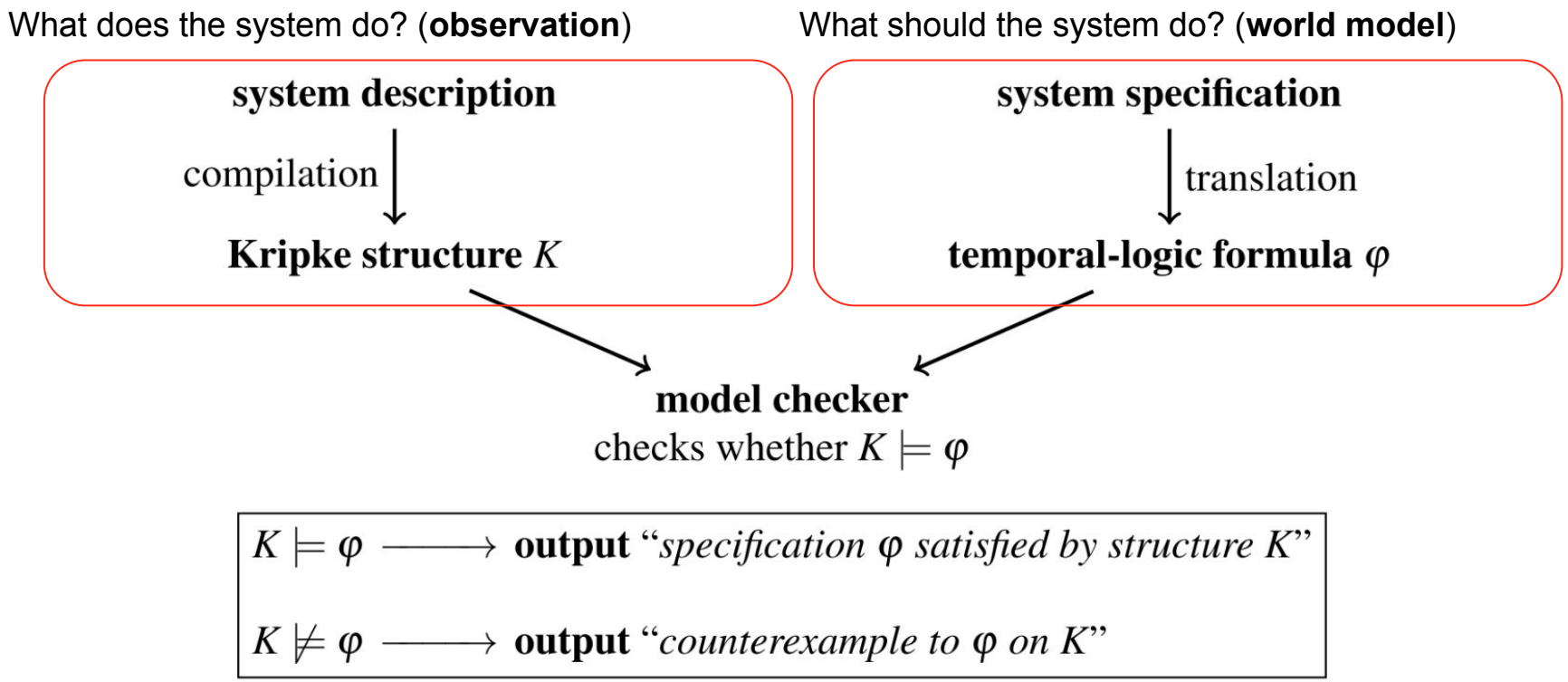
Model Checking is a group of software analysis techniques tailored to verify how a given a algorithm meets the provided specifications in an automated manner.



Source: Edmund M Clarke, Thomas A Henzinger, and Helmut Veith. "Introduction to model checking".

Visual Model Checking

What is model checking?



Source: Edmund M Clarke, Thomas A Henzinger, and Helmut Veith. "Introduction to model checking".

Agentic, Code-Based Image Analysis Can't Do Retrieval

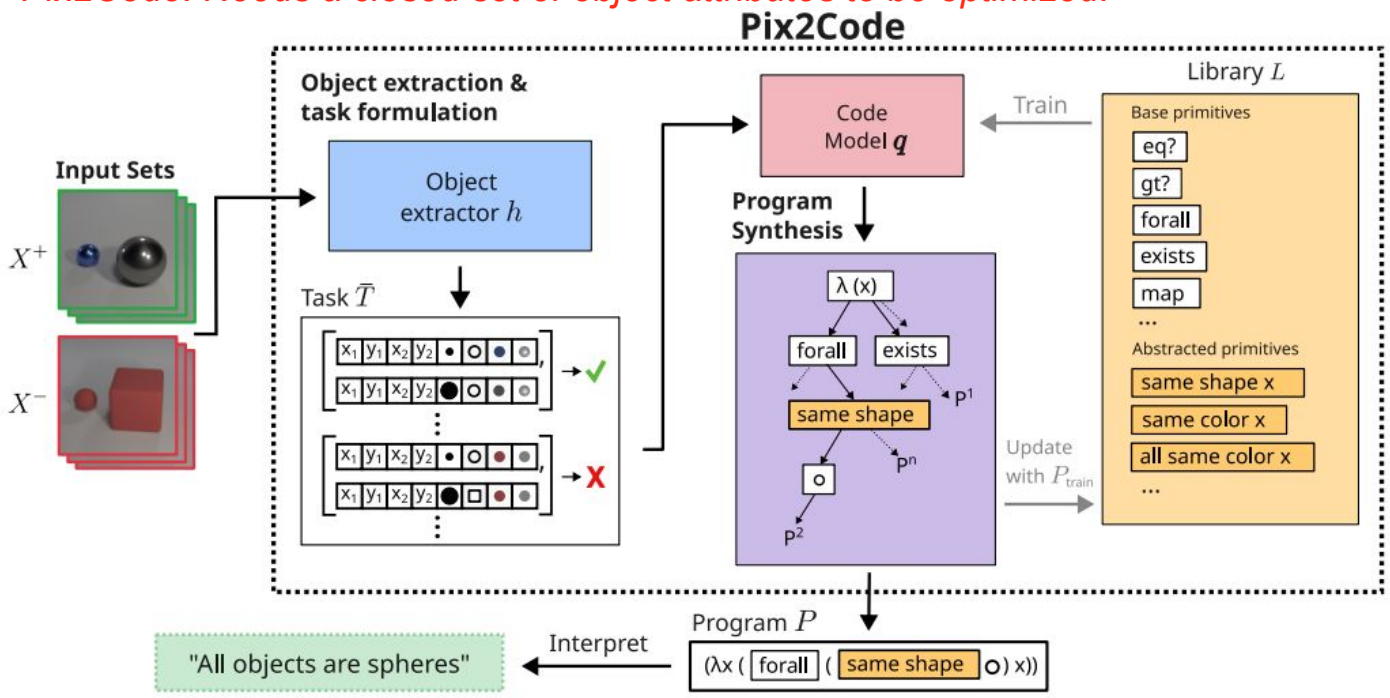
ViperGPT: VQA code generation with 1:1 Question-Answer ratio.

Query: How many muffins can each kid have for it to be fair?

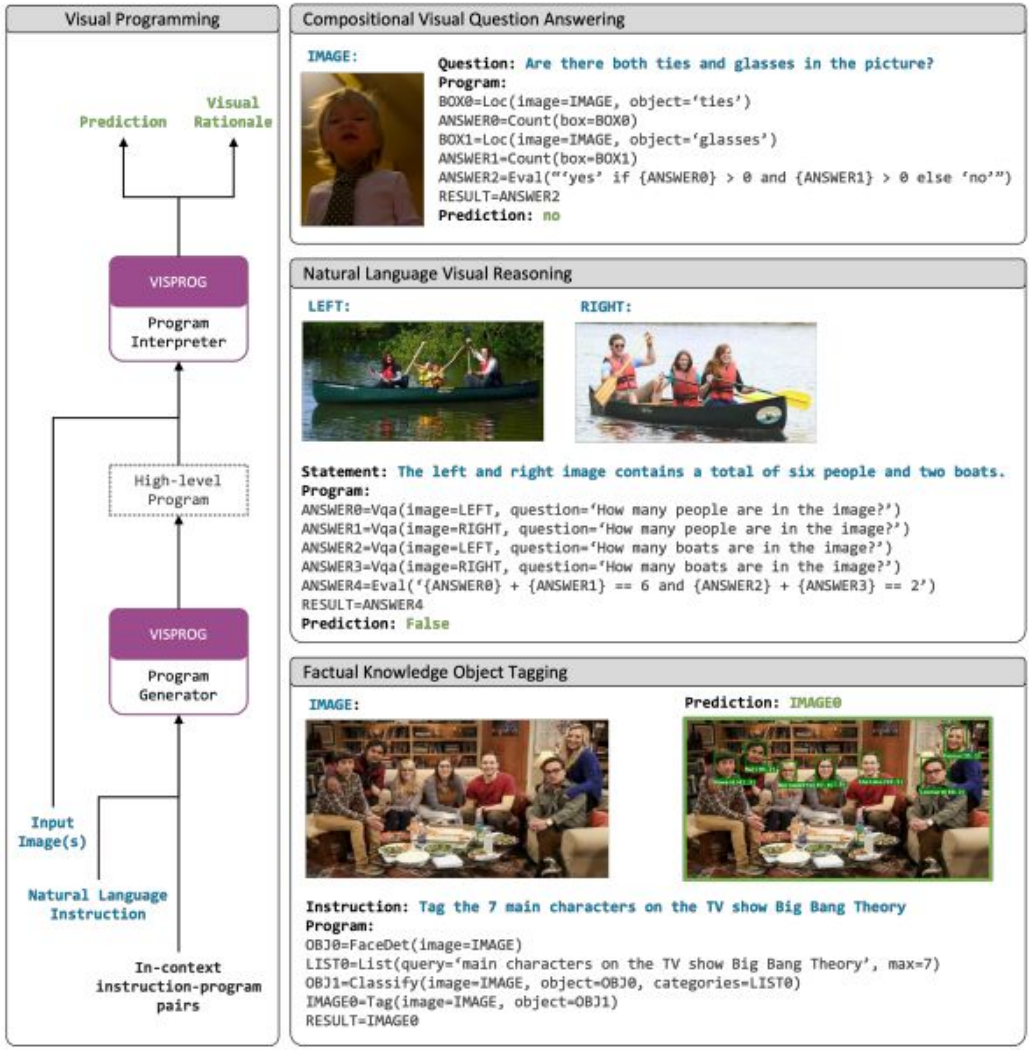
Generated Code

```
def execute_command(image):
    image_patch = ImagePatch(image)
    muffin_patches = image_patch.find("muffin")
    kid_patches = image_patch.find("kid")
    return str(len(muffin_patches) // len(kid_patches))
```

Pix2Code: Needs a closed set of object attributes to be optimized.



VisProg: Question-specific programs, no universal routines.



Agentic, Code-Based Image Analysis Can't Do Retrieval

Ours: The idea is to generate a visual routine with agnosticity to any image. This routine should be an abstraction for a broad semantical triplet which can be evaluated to True or False.

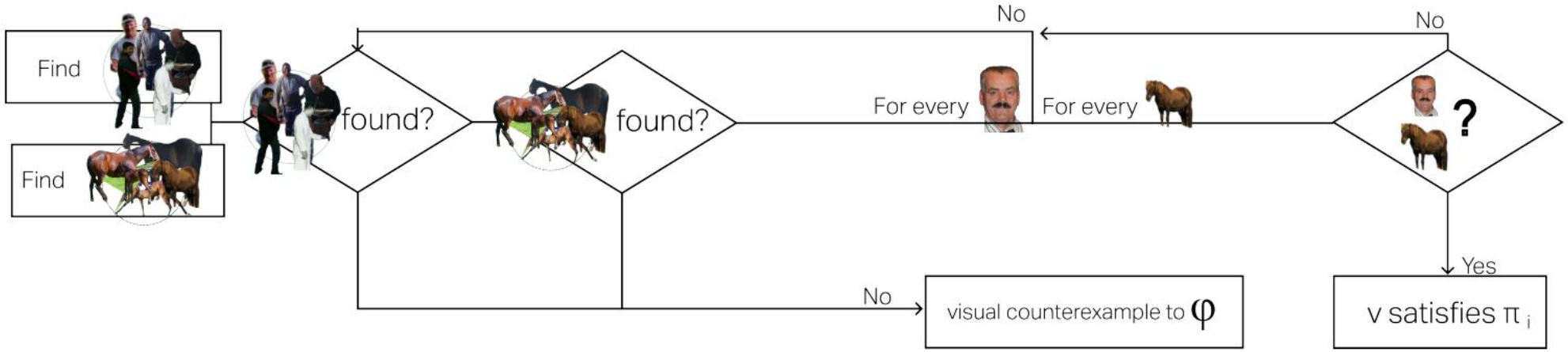


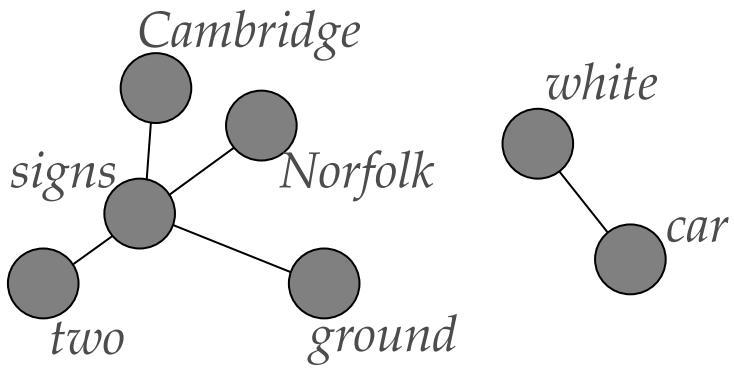
Fig. 3. Example of an actual visual routine for $(man, riding, horse)$, on which a program is synthesized to recognize an abstract statement with agnosticity of any specific image.

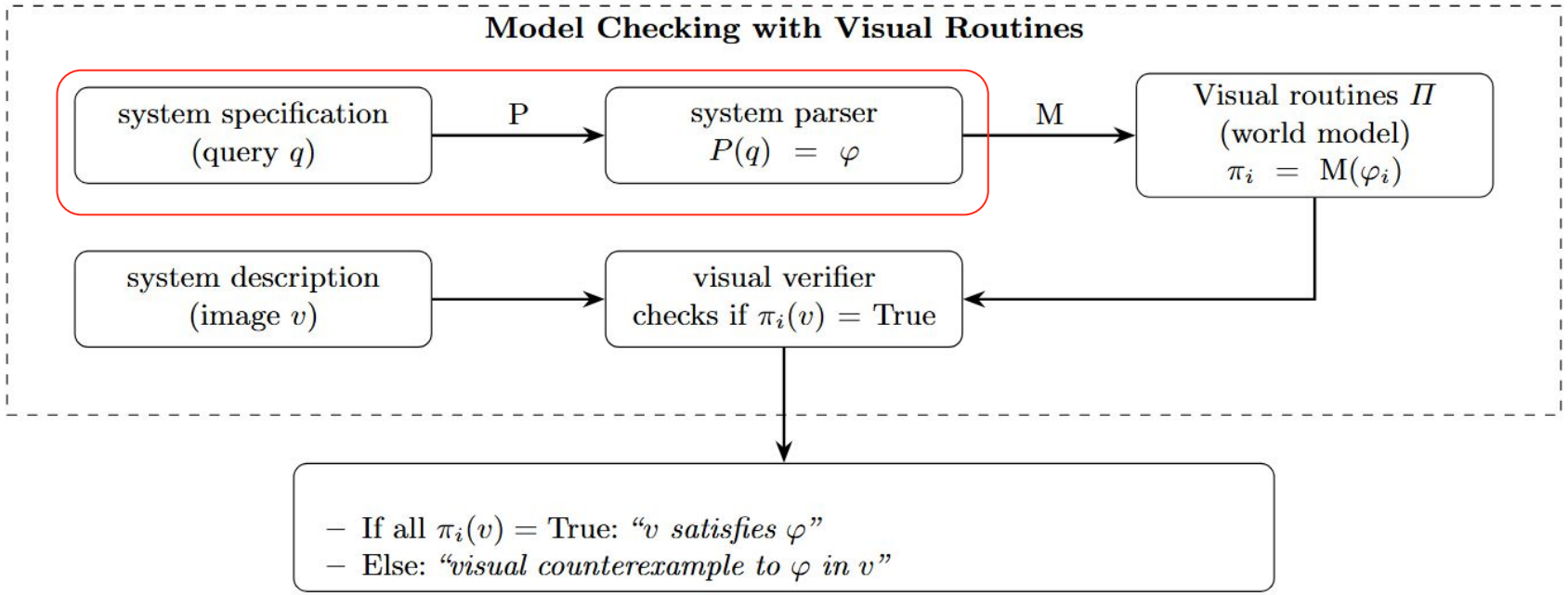
Visual Model Checking

Query:

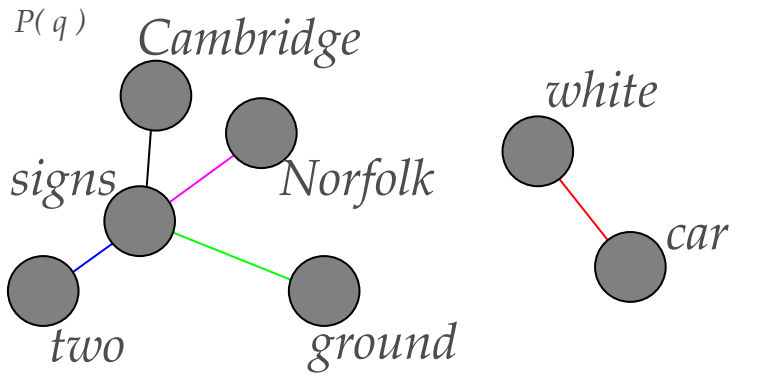
"A white car driving and two street signs on the ground, reading Cambridge and Norfolk."

$P(q)$

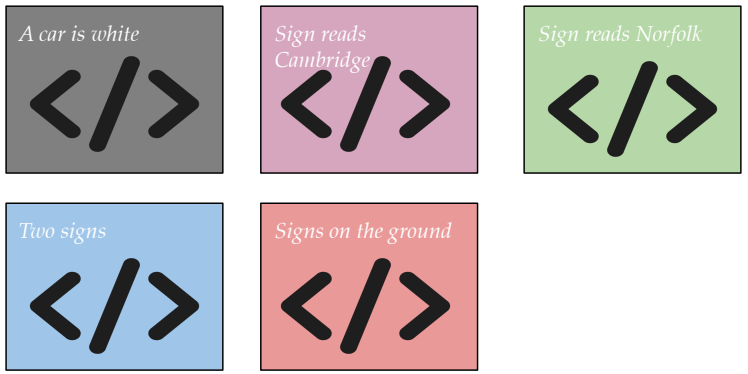


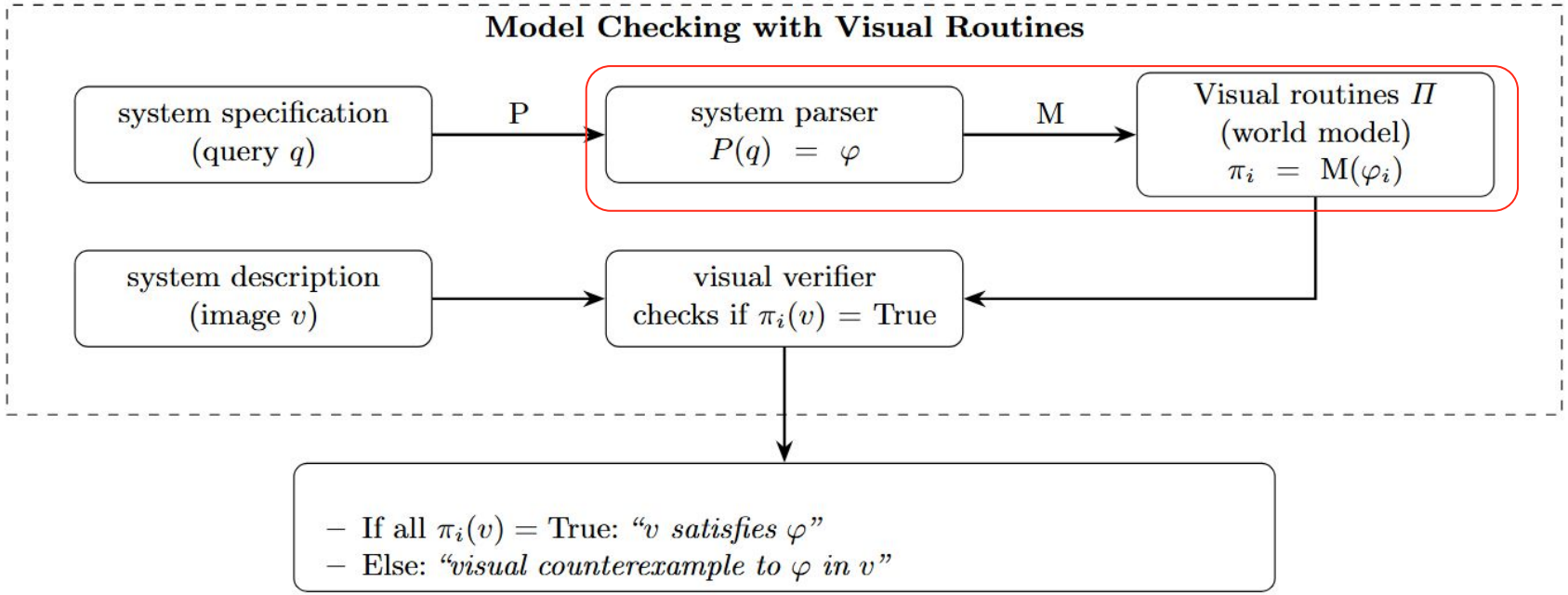


Visual Model Checking

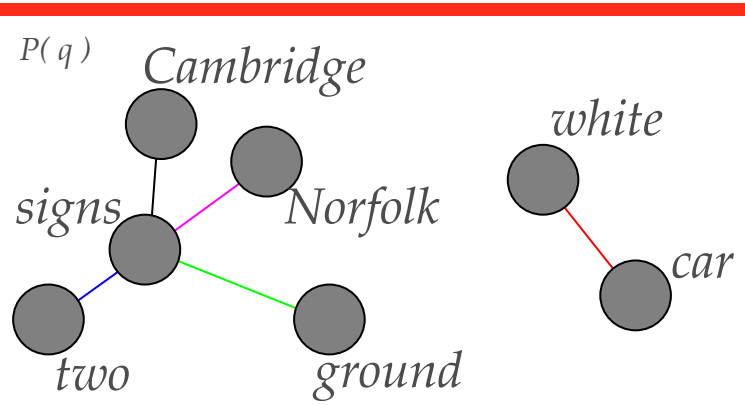


Visual routines: π

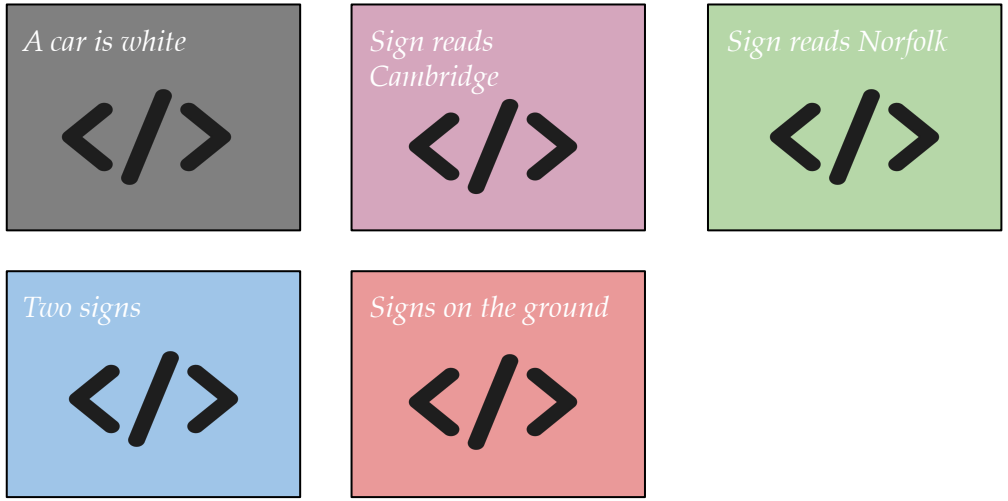


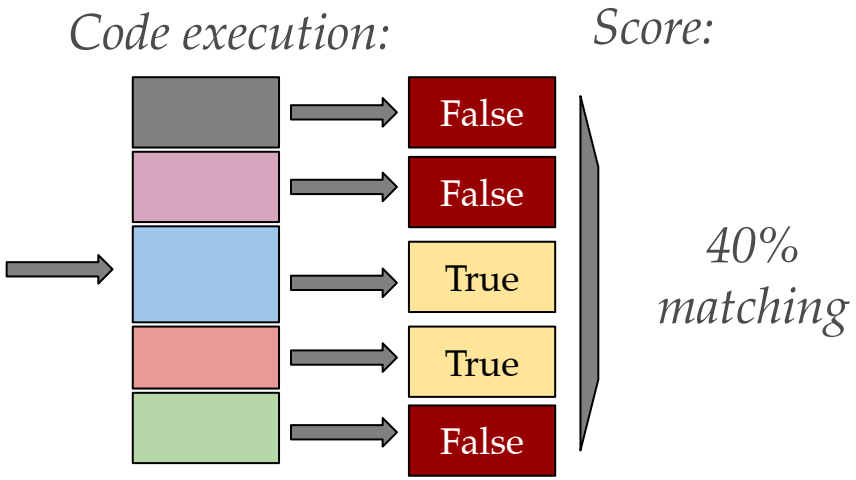


Visual Model Checking

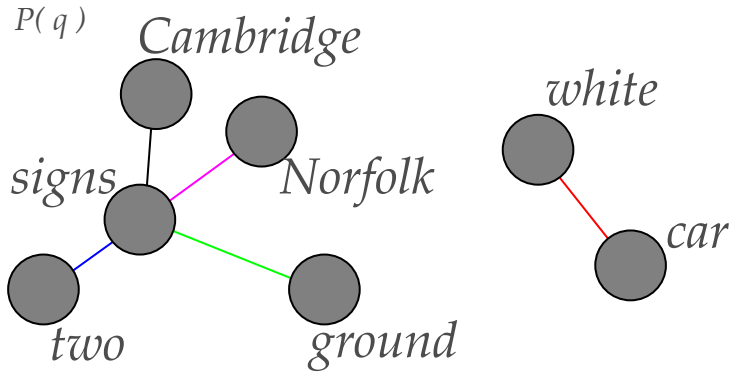


Visual routines: π

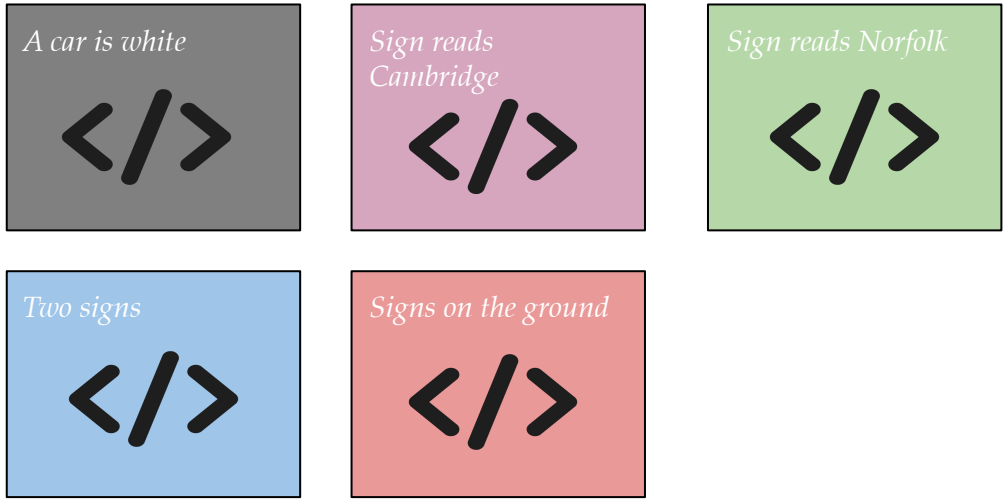




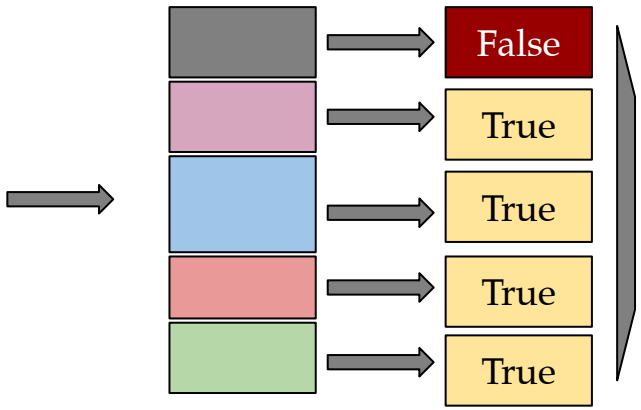
Visual Model Checking



Visual routines: π



Code execution:



Score:

80% matching

The Elephant in the Room

It is not *that* inefficient because in a closed vocabulary set-up the number of new visual routines drastically decreases over time (half the routines for COCO in only 50 queries).

Code executions @ T	0	12	24	36	50
Simulation #1	343	287	249	193	185
Simulation #2	342	285	225	197	184
Simulation #3	372	299	235	230	169

Evaluation Set-Up

The method is tailored for very specific kinds of high-stakes environments.

We are interested in query-image pairs which require enumeration, closer inspection, reading…

"The lunchbox has a cold sandwich, strawberry yogurt and orange juice."

…but some COCO captions are generic and unspecific.

A guy on a surf board in the water.

Evaluation Set-Up

Simple, unspecific captions (top 25% easiest)

	Easy			Combined			Hard			Zero-Shot?
Recall:	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	
CLIP (ViT-B) [17]	.486	.850	.892	.395	.710	.884	.152	.430	.565	✓
CLIP (ViT-H, QuickGELU) [10]	**.649**	.919	.973	.605	.899	.961	.283	.413	.674	✓
SigLIP (400M) [24]	.568	.811	.892	.581	.853	.946	.304	.478	.696	✓
BEIT-3 (COCO-Finetuned) [22]	.676	**.959**	.973	**.752**	**.944**	**.984**	**.435**	**.739**	.870	X
ALIGN [11]	.459	.469	.838	.481	.827	.938	.283	.413	.565	✓
(ours, base)	.432	.550	.784	.450	.700	.868	.196	.490	.739	✓
(ours) + CLIP	.592	**.959**	**.980**	.429	.638	.679	.043	.261	.565	✓
(ours) + BEIT	.551	.939	**.980**	.689	.913	.969	.391	.717	**.913**	X
(ours) + ALIGN	.469	.857	.878	.556	.872	.934	.261	.543	.717	✓

Evaluation Set-Up

Top 25% hardest, containing enumerations and fine-grained captions.

	Easy			Combined			Hard			Zero-Shot?
Recall:	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	
CLIP (ViT-B) [17]	.486	.850	.892	.395	.710	.884	.152	.430	.565	✓
CLIP (ViT-H, QuickGELU) [10]	**.649**	.919	.973	.605	.899	.961	.283	.413	.674	✓
SigLIP (400M) [24]	.568	.811	.892	.581	.853	.946	.304	.478	.696	✓
BEIT-3 (COCO-Finetuned) [22]	.676	**.959**	.973	**.752**	**.944**	**.984**	**.435**	**.739**	.870	X
ALIGN [11]	.459	.469	.838	.481	.827	.938	.283	.413	.565	✓
(ours, base)	.432	.550	.784	.450	.700	.868	.196	.490	.739	✓
(ours) + CLIP	.592	**.959**	**.980**	.429	.638	.679	.043	.261	.565	✓
(ours) + BEIT	.551	.939	**.980**	.689	.913	.969	.391	.717	**.913**	X
(ours) + ALIGN	.469	.857	.878	.556	.872	.934	.261	.543	.717	✓

Evaluation Set-Up

Our method performs slightly better on the zero-shot comparison. But fails to keep it up against fine-tuned models

	Easy			Combined			Hard			Zero-Shot?
Recall:	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	Rec@1	Rec@5	Rec@10	
CLIP (ViT-B) [17]	.486	.850	.892	.395	.710	.884	.152	.430	.565	✓
CLIP (ViT-H, QuickGELU) [10]	**.649**	.919	.973	.605	.899	.961	.283	.413	.674	✓
SigLIP (400M) [24]	.568	.811	.892	.581	.853	.946	.304	.478	.696	✓
[redacted]										
ALIGN [11]	.459	.469	.838	.481	.827	.938	.283	.413	.565	✓
(ours, base)	.432	.550	.784	.450	.700	.868	.196	.490	.739	✓
(ours) + CLIP	.592	**.959**	**.980**	.429	.638	.679	.043	.261	.565	✓
[redacted]										
(ours) + ALIGN	.469	.857	.878	.556	.872	.934	.261	.543	.717	✓

Re-Ranking Results

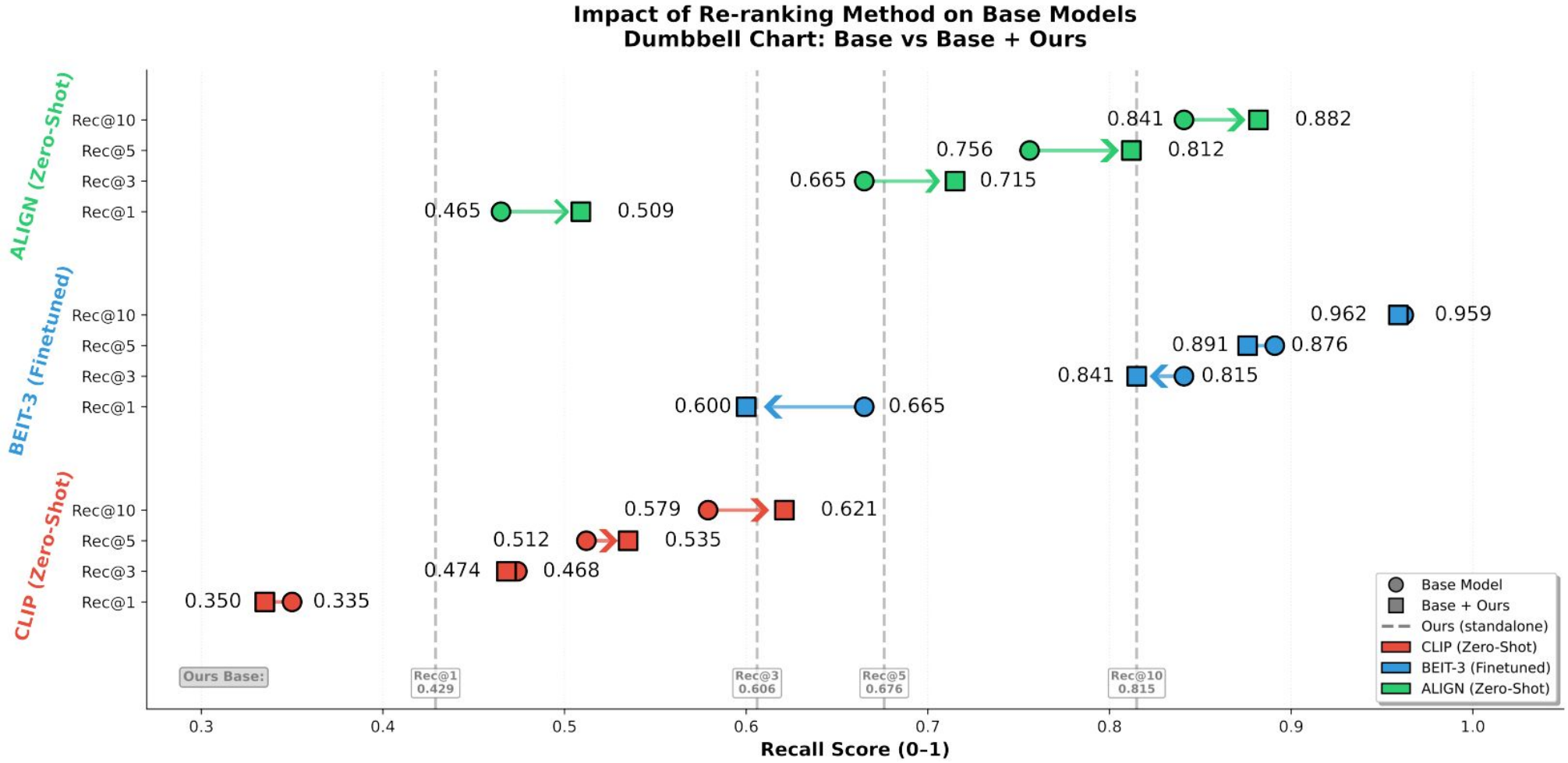
However, in the zero-shot scenario, the model is an excellent re-ranker.

	COCO-All				Zero-Shot?
Recall:	Rec@1	Rec@3	Rec@5	Rec@10	
CLIP (ViT-B) [17]	.350	**.474**	.512	.579	✓
CLIP + (ours)	**.355**	.468	**.535**	**.621**	✓
ALIGN [11]	.465	.665	.756	.841	✓
ALIGN + (ours)	**.509**	**.715**	**.812**	**.882**	✓
BEIT-3 (COCO-Finetuned) [22]	**.665**	**.841**	**.891**	**.962**	X
BEIT + (ours)	.600	.815	.876	.959	X
(ours, base)	.429	.606	.676	.815	✓

Table 2. Can our method be used as a re-ranker of traditional embedding-based approaches? **We highlight those situations on which the method acts as a booster** to the original performance. We observe how the LLM common-sense knowledge helps in zero-shot scenarios. In fine-tuned embeddings (BEIT-3), the drop in performance is not very significative.

Re-Ranking Results

However, in the zero-shot scenario, the model is an excellent re-ranker.



Qualitative Results

Triplet ($\varphi_i \in \varphi$)	Routine ($\pi_i \in \Pi$)	Groundtruth Image (v_{gt})	$\pi_i(v_{gt})$	Error Type
bench $\xrightarrow{\text{located by}}$ shore	Code 1.5		True	–
woman $\xrightarrow{\text{posing}}$ picture	Code 1.6		False	Bad triplet
lake $\xrightarrow{\text{with}}$ two boats	Code 1.7		False	Annotation is wrong
road $\xrightarrow{\text{made of}}$ dirt	Code 1.8		False	Bad code execution
sign $\xrightarrow{\text{reads}}$ Norfolk	Code 1.9		True	–
sign $\xrightarrow{\text{reads}}$ Cambridge	Code 1.10		True	–
snow $\xrightarrow{\text{is}}$ under	Code 1.11		False	Bad triplet
bathtub $\xrightarrow{\text{is}}$ white	Code 1.12		False	Bad code generation

```python
def execute_command(image: ImagePatch):
    api = DescriptiveImageAPI()

    bathtubs = api.find_objects(image, 'bathtub')
    print(f"Found {len(bathtubs)} bathtub(s).")

    if not bathtubs:
        return False

    is_white = bathtubs[0].pil_image.getcolors(
        maxcolors=2)
    print(f"Is the bathtub white? {'Yes' if
        is_white else 'No'}")

    return is_white
```

Conclusions

Reading text and Symbols

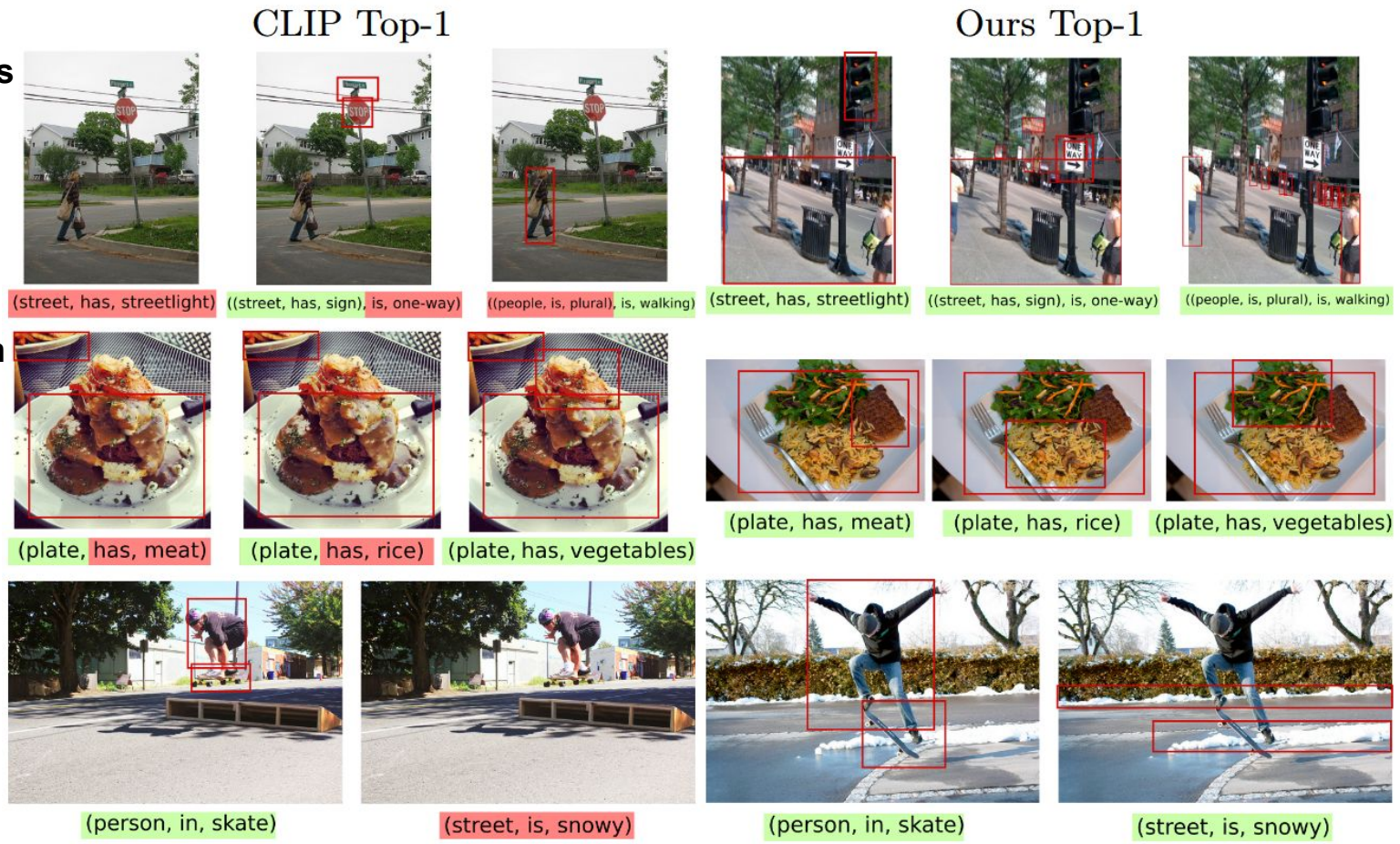
The model has the capacity to focus on small pieces of information and extract the text regardless of (A) the resolution or (B) the saliency of the object in the image.

Counting and Enumeration

The model can count step-by-step, hence bypassing the spurious correlations emerging from the pattern recognition capabilities of CLIP.

Attribute Dominance

Because a visual routine is generated at the most local level, attribute dominance is not a problem anymore.



Future Work

- The used LLM is Phi-4 for both graph and code generation, we are certain that with state-of-the-art systems the results would drastically improve. Zero prompt engineering was done, maybe the results could improve in this sense; but we wanted to keep the contribution more or less model-agnostic.

- The system offers perfect explainability and tracing, while keeping on-par performance on retrieval benchmarks.

- While the system proofs worthy on images analysis with compositionality, counting, arithmetic, reading, etc. Internal tests on archival data (containing named entities and factual information) shows the limitation of the method, as off-the-shelf detectors and recognition systems do not handle well the identification of named entities in images (buildings, people, institutions…)

Computer
Vision Center

Edifici O, Campus UAB
08193 Bellaterra
(Cerdanyola)
Barcelona, Spain
www.cvc.uab.es